UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Fisica e Astronomia "Galileo Galilei"

Corso di Laurea in Fisica

Tesi di Laurea

Studio di procedure di minimizzazione con gradiente deformato

Relatore

Prof./Dr. Marco Baiesi

Laureando

Dario Cariolato

Anno Accademico 2019/2020

Prefazione

Perché studiare algoritmi di minimizzazione

Un algoritmo di minimizzazione è una procedura tramite la quale, attraverso una serie di passaggi, si riesce ad arrivare al minimo di una funzione. Essi sono alla base del machine learning, strumento di sempre più utilizzo e interesse in vari campi (compresa la fisica); dunque la spinta moderna allo sviluppo di questa disciplina promuove la ricerca di algoritmi più rapidi e maggiormente stabili.

In questa tesi parlerò appunto della velocità di alcuni algoritmi di minimizzazione. Se questo intento dovesse realizzarsi si aprirebbe al campo delle reti neurali una nuova possibilità per rendere la procedura di apprendimento sempre più breve.

Perché parlare di gradiente trasformato

Una delle possibilità più interessanti per un algoritmo di minimizzazione è quella di sfruttare il gradiente della funzione (cambiato di segno). Seguendo in effetti questo campo vettoriale, è noto che si riesca a raggiungere il minimo della funzione.

Considerato quindi un algoritmo di questa tipologia è possibile sostituire il campo vettoriale comunemente usato, il gradiente, con un altro. Attuando questo scambio ci si accorge che in generale il gradiente non è ottimale, cioè non è il campo attraverso il quale l'algoritmo porta al minimo della funzione con il minor sforzo computazionale.

Questo è particolarmente vero nel caso in cui si abbiano delle valli di stabilità (cioè delle configurazioni come in figura 1) particolarmente lunghe, poiché sui pendii il gradiente punta verso il fondo della valle invece di puntare direttamente al minimo. Si noti che gli algoritmi, in generale, sfruttano la direzione del gradiente come indicatore della posizione del minimo.

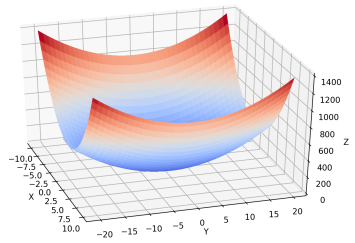


Fig.1: Esempio di valle di stabilità

Ciò che vogliamo è quindi un campo vettoriale che renda l'algoritmo di minimizzazione più veloce. Dato che la procedura deve valere in generale per ogni funzione, sembra una buona idea partire dal campo vettoriale che già conosciamo, il gradiente, e di trasformarlo. Inoltre funzionando già esso all'interno dell'algoritmo, questa trasformazione non dovrà cambiare troppo la natura del nostro campo.

In dettaglio la proposta di trasformazione è:

$$Grad_T = sgn(Grad) \times |Grad|^H \qquad \text{con} \quad 0 \leq H \leq 1$$

dove il modulo, la funzione segno e l'elevazione a potenza sono rispettivamente applicate ad ogni componente del nostro gradiente. [1]

L'obiettivo di questa tesi è quello di studiare e caratterizzare $Grad_T$ per una funzione semplice, un paraboloide ($\mathbb{R}^2 \Rightarrow \mathbb{R}$), e per un algoritmo che di fatto segue le curve integrali del campo, in maniera però discreta.

Come prima cosa ho quindi realizzato un programma in Python che eseguisse l'algoritmo di minimizzazione. L'algoritmo di cui sto parlando è molto semplice, ne descriverò ora il funzionamento.

Supponiamo esso consideri il campo vettoriale v e parta da una certa coordinata (X, Y), allora il nostro algoritmo considererà il punto nella direzione indicata da $v(X, Y)$ ad una distanza prefissata infinitesima p. Questo secondo punto considerato (X', Y') andrà a sostituire il punto (X, Y) e verranno eseguite le stesse operazioni sopra descritte (si calcolerà $v(X', Y')$ e si considererà il punto a distanza p in tale direzione). L'algoritmo seguirà quindi questo percorso fino a che $|(X, Y)| \leq p$, dato che il minimo di un paraboloide si trova in coordinata $(0, 0)$ (dove (X, Y) è il punto considerato in quel momento).

L'algoritmo inoltre calcolerà la distanza percorsa come la somma di tutti i passi di dimensione p eseguiti. Sarà appunto questa distanza il termine di confronto per valutare di quanto un campo vettoriale possa essere migliore di un altro, equivalendo minore distanza a minore quantità di passi e quindi minor sforzo computazionale.

Studio di $Grad_T$

Prima di vedere cosa avviene effettivamente nell'esecuzione del nostro algoritmo utilizzando un campo vettoriale rispetto ad un altro, vediamo cosa accade ad un vettore qualsiasi soggetto alla trasformazione di interesse.

Osserviamo innanzitutto che la prima parte della deformazione prende il modulo di tutte le componenti del vettore, cioè lo proietta nel primo quadrante del sistema di riferimento. La seconda operazione è invece l'elevazione a potenza, della quale parleremo in seguito. Infine viene riportato il vettore nel suo quadrante originario tramite l'operazione segno.
Ciò che risulta evidente da questo procedimento è che studiare l'operazione di elevazione a potenza nel primo quadrante è sufficiente per caratterizzare completamente la nostra trasformazione.

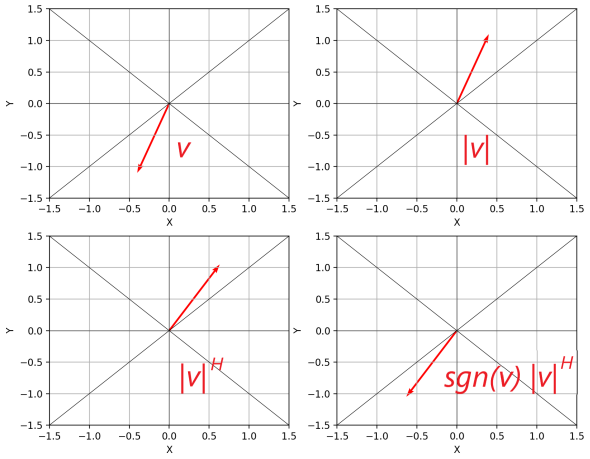


Fig.2: Trasformazione di un vettore generico v con $H = 0.5$

Parliamo ora quindi dell'elevazione a potenza di un vettore nel primo quadrante. Di tale operazione non ci interessa la variazione del modulo del vettore dato che il nostro algoritmo non lo considera nel suo svolgimento (tutti i vettori sono in pratica normalizzati a p). La parte di trasformazione importante è la rotazione del vettore.

Studiamo quindi come cambia l'angolo tra il vettore e l'asse X tramite la tangente, funzione monotona dell'angolo. Esprimo quindi, di seguito, le relazioni utili a tale scopo:

$$Tg(\theta) = \frac{Y}{X} \quad , \quad Tg(\theta') = \frac{Y'}{X'} \quad \text{e} \quad \frac{Y'}{X'} = (\frac{Y}{X})^H$$

dove θ, X e Y si riferiscono al vettore non trasformato mentre θ', X' e Y' al vettore trasformato.

Si osservano comportamenti diversi in base alla zona in cui si trova il vettore da trasformare. Possiamo dividere in particolare le zone in $\frac{X}{Y} > 1$ (Zona 1), $\frac{X}{Y} < 1$ (Zona 2) e $\frac{X}{Y} = 1$ (Bisettrice).
Si trovano in maniera ovvia le seguenti relazioni:

- $\frac{X}{Y} > \frac{X'}{Y'}$ in Zona 1
- $\frac{X}{Y} < \frac{X'}{Y'}$ in Zona 2
- $\frac{X}{Y} = \frac{X'}{Y'}$ sulla Bisettrice

(Ricordando $0 \leq H \leq 1$ ed escludendo $H = 1$ trasformazione identità)

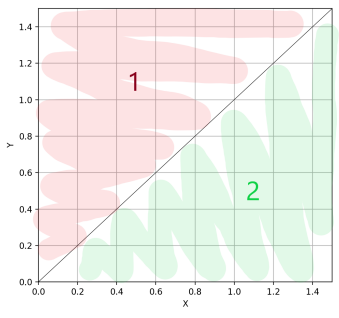


Fig.3: Rappresentazione delle zone

Relazioni che si traducono nel seguente modo $\frac{X}{Y} > \frac{X'}{Y'} \Rightarrow Tg(\theta) > Tg(\theta') \Rightarrow \theta > \theta'$ in relazioni equivalenti per gli angoli dei vettori corrispondenti.

Esplicando a parole quanto abbiamo appena scoperto, potremmo dire che la nostra trasformazione complessivamente ruota un vettore avvicinandolo alla bisettrice del proprio quadrante (come affermato in precedenza con ovvia estensione ai quadranti diversi dal primo).

Vorrei aggiungere inoltre, prima di concludere la sezione, tre osservazioni:

1. Innanzitutto riconosciamo che si hanno tre "punti fissi"[1]: la bisettrice e i due assi. In particolare, anche se la tangente non è definita per un vettore con X = 0, il comportamento è totalmente prevedibile: la direzione del vettore trasformato è la stessa del vettore di partenza.

2. In secondo luogo, se avessimo considerato $H > 1$, sarebbe stato facile verificare la rotazione del nostro vettore nella direzione opposta, distanziandosi dalla bisettrice. Si sarebbe però avvicinato agli assi X e Y che, come osservato prima, sono anch'essi "punti fissi".

3. Infine è interessante vedere che l'angolo trasformato non dipende dalla norma del vettore di partenza. Valendo infatti $Tg(\theta') = (\frac{Y}{X})^H$ avremo che un riscalamento $(X, Y) \Rightarrow (aX, aY)$ produce lo stesso valore θ' di arrivo.

Paraboloide lungo l'asse Y

É stato quindi considerato il paraboloide di equazione $L_1X^2 + L_2Y^2 = Z$. Si è deciso di utilizzare questa funzione poiché in prima approssimazione (approssimazione in serie di Taylor) ogni altra funzione ci assomiglia, considerando un intorno sufficientemente piccolo del minimo.

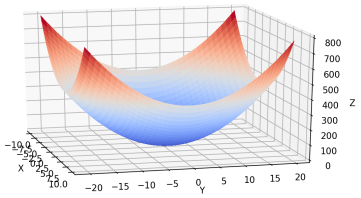


Fig.4: Esempio di paraboloide con $L_1 = 4$ $L_2 = 1$

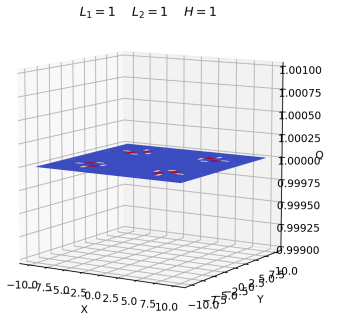
Sono partito da $L_1 = 1$ $L_2 = 1$, il caso più semplice, ed ho impostato l'algoritmo perché funzionasse prima con il campo non trasformato ($H = 1$) e poi trasformato con $H = 0.5$; l'ho dunque eseguito su una griglia nel piano XY, la quale era formata da punti con coordinata $-10 \leq X \leq 10$ e spaziatura di 0.25 tra un punto e un altro (nella stessa coordinata) e con coordinata $-10 \leq Y \leq 10$ e spaziatura di 0.25 tra un punto e un altro (nella stessa coordinata).
I risultati sono stati una serie di percorsi, ciascuno corrispondente alle sue coordinate X e Y di partenza.

Gli esiti sono espressi da grafici tridimensionali nei quali è rappresenta la seguente variabile in coordinata Z:

$$Q = \frac{percorso(X, Y)}{\sqrt{X^2 + Y^2}}$$

dove percorso(X, Y) indica la quantità calcolata dall'algoritmo nel punto corrispondente (X, Y) e notando che il termine al denominatore non è altro che la distanza di tale punto dal minimo.
Questo perché, se avessimo messo in coordinata Z il valore del percorso calcolato dall'algoritmo, non avremmo potuto ricavare alcun significato immediato da tale grafico, se non quello per cui all'aumentare della distanza dal minimo del punto di partenza del percorso sarebbe conseguentemente aumentato il percorso necessario per arrivarci.
Inoltre questo metodo ci dà un termine di paragone su cui fare alcune valutazioni, per esempio se si dovesse trovare che in un certo punto il grafico assume valore $Q = 1$, avremmo che tale percorso è ideale, ovvero esso coincide con la distanza del punto dal minimo e dunque si tratta di quello più breve.

[1]Non sono esattamente punti fissi in quanto cambia il modulo del vettore, ma li possiamo considerare tali per l'algoritmo utilizzato.



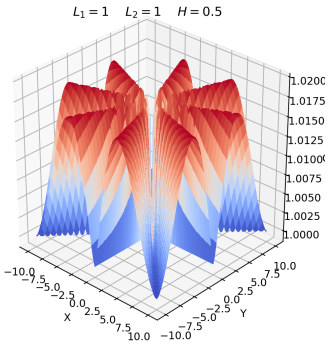


Fig.5: Q-plot

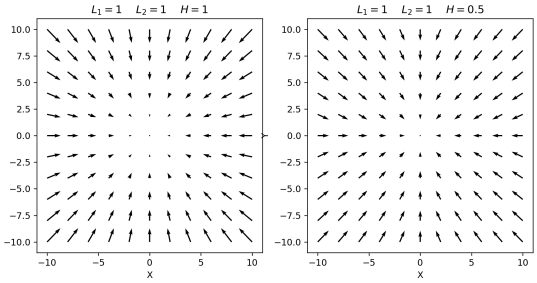
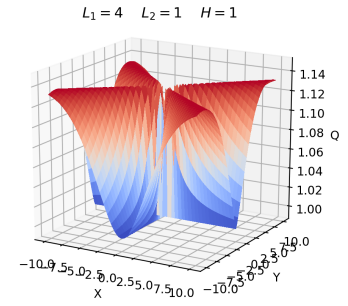


Fig.6: Gradiente e trasformazione

Confrontando allora i percorsi calcolati con $H = 1$ e $H = 0.5$ si trova che in questo primo caso non solo è conveniente utilizzare H = 1 ma è anche ideale ($Q = 1$ sempre). Ciò risulta evidente guardando il gradiente della funzione, dato che tutti i vettori puntano direttamente al minimo. Il gradiente trasformato è invece distorto e quindi non possiede esattamente tale comportamento.

Ciò che ci interessa però è la situazione nella quale ci sono valli di stabilità lunghe; vediamo quindi cosa accade in tale caso. Rappresento questa condizione lasciando $L_2 = 1$ e variando $L_1 = 2, 3, 4, 5, 6$. Si mostra in figura come modello esemplificativo $L_1 = 4$:



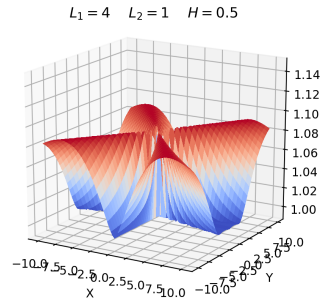


Fig.7: Q-plot

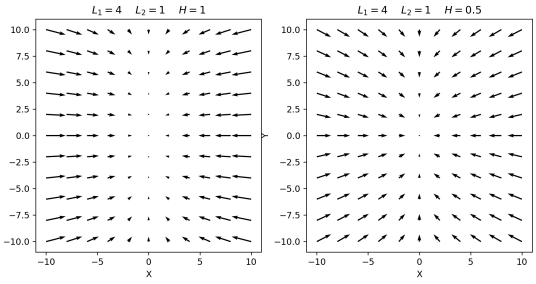


Fig.8: Gradiente e trasformazione

Ci si accorge non solo che ora H = 1 non è più ideale ovunque, ma che nei punti in cui non lo è, cioè lungo le bisettrici del sistema di assi XY, il campo con H = 0.5 ha una performance migliore. Se ne può dedurre la ragione ancora una volta guardando i due campi vettoriali: nelle zone lungo la bisettrice il gradiente tende a raggiungere come prima cosa la valle per poi svoltare fino al minimo; la trasformazione invece, ruotando il gradiente nella direzione del minimo, focalizza i percorsi perché siano più diretti.

Si definisca dunque $< Q >$ come media tra tutti i valori di Q calcolati nei punti della griglia sul piano XY e corrispondenti ad un certo paraboloide.

Nella seguente tabella sono presentati i valori di $< Q >$ delle funzioni di cui abbiamo precedentemente parlato:

	$L_1 = 1$	$L_1 = 2$	$L_1 = 3$	$L_1 = 4$	$L_1 = 5$	$L_1 = 6$
$H = 1$	1.00	1.03	1.06	1.09	1.11	1.12
$H = 0.5$	1.01	1.02	1.04	1.05	1.06	1.07

(Ricordo $L_2 = 1$ nelle funzioni da noi considerate).
Non solo quindi H = 0.5 è migliore, ma è tanto migliore quanto più lunga è la valle.

Specifico che l'algoritmo è stato impostato perché funzionasse con $p = 0.001$, dove p non è altro che la distanza predefinita di spostamento nella direzione del campo vettoriale. É per questa ragione che si sono arrotondati i valori in tabella alla seconda cifra decimale, essendo le cifre successive incerte.

La scala

Nella sezione precedente abbiamo analizzato cosa accade per alcuni valori specifici di L_1 e L_2. Mi sono quindi chiesto cosa sarebbe successo se avessi moltiplicato L_1 e L_2 per uno stesso numero, cioè cosa accadrebbe se mantenessi il rapporto tra i due valori ma li cambiassi singolarmente.

Ho dunque eseguito l'algoritmo per $L_1 = 4$ e $L_2 = 1$ entrambi moltiplicati rispettivamente da uno stesso valore $I = 0.25, 0.5, 1, 2, 3, 4, 5, 6$ e per $H = 1$, $H = 0.5$.

Ciò che ci si aspetta è che il grafico sia sempre lo stesso anche per valori di I diversi. Questo perché, se osserviamo la forma del gradiente $Grad = (2L_1X, 2L_2, Y)$, si trova che $Grad' = I \times Grad$, dove $Grad'$ è il gradiente con L_1 e L_2 moltiplicati per I. Perciò avremo solo un riscalamento del gradiente che non porta alcuna modifica nell'algoritmo per qualsiasi valore di H.

L'asse Z dei grafici in figura 9 esprime il seguente valore $D = Q_{0.25} - Q_6$, dove il pedice corrisponde al valore assunto da I nel calcolo di Q. Essendo nei due grafici $D = 0$ sempre, in effetti si trova è che i percorsi calcolati sono identici.
Questa analisi giustifica quindi la validità delle nostre precedenti affermazioni per qualunque scala della funzione.

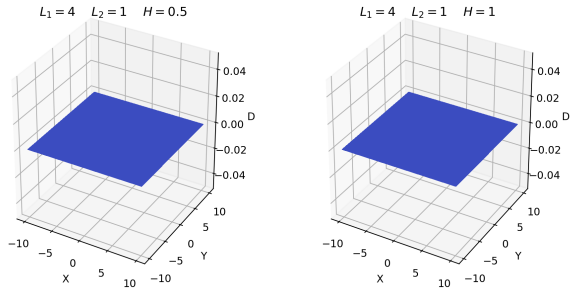


Fig.9: D-plot

Vorrei inoltre osservare che l'equivalenza della nostra analisi nel caso in cui si voglia attuare una trasformazione di scala ai due assi è chiara: $\begin{cases} X' = a \times X \\ Y' = a \times Y \end{cases}$

Infine specifico che l'algoritmo, questa volta, è stato impostato perché funzionasse con $p = 0.1$ molto meno impegnativo computazionalmente per il mio computer, ma sufficiente per un analisi qualitativa come quella appena fatta.

Paraboloide lungo un asse qualsiasi

Ho poi proceduto ad analizzare cosa sarebbe capitato se avessi ruotato gli assi del sistema di riferimento.

La legge di trasformazione per un campo vettoriale è $v(x) = v'(x')$, dove in particolare per una rotazione del sistema di riferimento x' e v' sono dati dalla rotazione stessa. Questo significa che la trasformazione è separabile in due parti: la prima riguarda la posizione del vettore (il vettore viene infatti posizionato in x' e non più in x), la seconda invece riguarda il vettore stesso, trasformazione che in questo caso è proprio una rotazione.

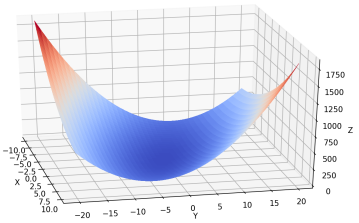


Fig.10: Esempio paraboloide $L_1 = 4$ $L_2 = 1$ ruotato di $\theta = 0.25\pi$

La prima parte della trasformazione è evidentemente poco interessante, infatti avrà come conseguenza semplicemente la ricollocazione nel piano XY dei percorsi.

La seconda parte della trasformazione è invece importante: infatti, come sappiamo, il nostro algoritmo agisce diversamente in base all'angolo che il vettore forma con gli assi del sistema di riferimento. In particolare è il risultato dell'elevazione a potenza a differire, quindi nel caso $H = 1$ i percorsi mantengono gli stessi valori relativamente a questo aspetto.

Ho quindi ruotato gli assi con la trasformazione: $\begin{cases} X' = cos(\theta)X - sin(\theta)Y \\ Y' = sin(\theta)X + cos(\theta)Y \end{cases}$

che dà luogo alla funzione $L_1(cos(\theta)X' + sin(\theta)Y')^2 + L_2(cos(\theta)Y' - sin(\theta)X')^2 = Z$.

Gli angoli di rotazione sono stati $\theta = 0.17\pi, 0.26\pi, 0.3\pi, 0.5\pi, 0.67\pi, 0.75\pi, 0.8\pi, \pi$ ed è quindi stato eseguito l'algoritmo con $L_1 = 4$, $L_2 = 1$, $H = 1$ e $H = 0.5$.

Innanzitutto si osserva che, come previsto, l'unico cambiamento nel caso $H = 1$ è la rotazione dei percorsi precedentemente calcolati. Inoltre si nota che per $H = 0.5$ vi è una periodicità per rotazioni di $\theta = 0.5\pi$. Come possiamo vedere in figura 11 infatti il calcolo eseguito dall'algoritmo è identico salvo una rotazione appunto di 0.5π della posizione dei percorsi.

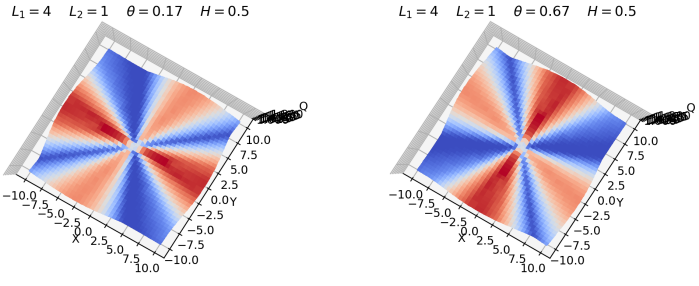


Fig.11: Q-plot

Questo risultato, se pur non immediato, deriva dalla struttura della trasformazione. Nella figura 12 la prima zona colorata di rosso a partire da destra ruotata di 0.5π diventa la seconda e lo stesso vale per le due zone verdi. Sapendo ora che il punto di attrazione del secondo quadrante è il punto di attrazione del primo quadrante ruotato della medesima quantità (i punti di attrazione corrispondono alle due bisettrici), ci risulta naturale assumere la struttura periodica affrontata precedentemente.

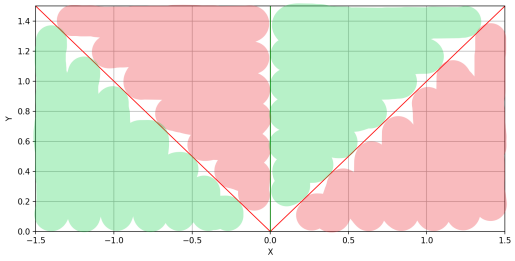


Fig.12: Quadranti sel sistema di riferimento

Ci limiteremo quindi da ora ad analizzare le funzioni con angoli di rotazione pari a $\theta = 0, 0.17\pi, 0.25\pi, 0.3\pi$. Considero innanzitutto $\theta = 0.25\pi$, producendo i seguenti grafici:

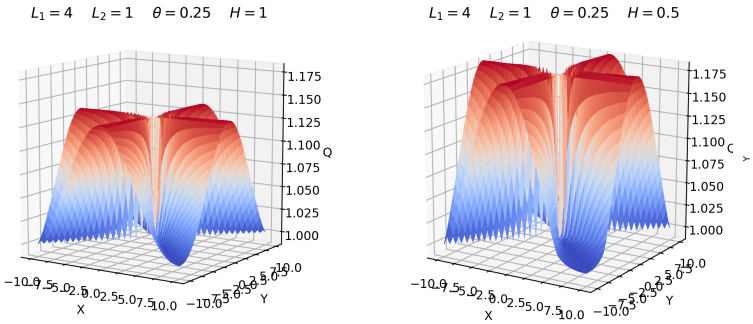


Fig.13: Q-plot

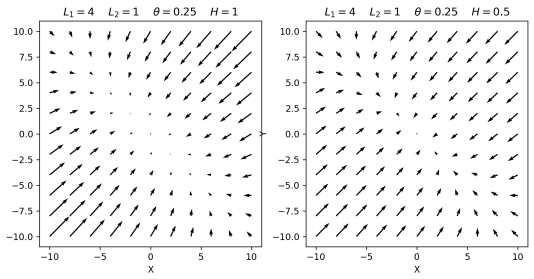


Fig.14: Gradiente e trasformazione

Ci si accorge che $H = 1$ è ora migliore di $H = 0.5$. Prevedibile perché, come abbiamo visto precedentemente, il gradiente trasformato migliora i percorsi lungo le bisettrici del sistema di riferimento che in questo caso sono già ideali; peggiorano invece le zone lungo gli assi (zone poco ideali già con $H = 1$). Guardando attentamente anche il campo vettoriale si riscontra che quello con $H = 0.5$ ha una tendenza maggiormente decentrante rispetto a $H = 1$.

Da queste considerazioni si può arrivare alla conclusione che i valori $\theta = 0.17\pi, 0.3\pi$ siano caratterizzabili come stati intermedi tra le condizioni di miglioramento $\theta = 0, 0.5\pi$ e quella di peggioramento $\theta = 0.25\pi$.

Si ottiene quindi la seguente tabella di $< Q >$ per la funzione con $L_1 = 4$ e $L_2 = 1$:

	$\theta = 0$	$\theta = 0.17$	$\theta = 0.25$	$\theta = 0.3$
$H = 1$	1.09	1.07	1.07	1.07
$H = 0.5$	1.05	1.09	1.09	1.09

Emerge come anche nei punti intermedi sia peggiore $H = 0.5$ rispetto a $H = 1$ e ciò non fa sperare in maniera positiva per il comportamento generale medio su tutti gli angoli.

Infine preciso che nei primi grafici in cui si sono confrontati i valori di $\theta = 0.17\pi, 0.26\pi, 0.3\pi, 0.5\pi, 0.67\pi, 0.75\pi, 0.8\pi, \pi$ l'algoritmo è stato eseguito con passo $p = 0.1$, sempre per limitare lo sforzo computazionale.
Successivamente per lo studio di $\theta = 0.17\pi, 0.26\pi, 0.3\pi$ esso è invece stato eseguito con passo $p = 0.001$ per ottenere una maggiore precisione.

Il set di punti di partenza

É possibile notare nei grafici raffigurati fino a questo momento, rappresentanti i percorsi normalizzati alla distanza, una certa simmetria. Il valore di Q per un certo punto nel piano XY è infatti uguale a tutti i valori, indicati con un sistema di coordinate circolari, aventi stesso angolo e raggio qualsiasi.
L'esecuzione dell'algoritmo per tutti i punti della griglia con un passo piccolo come quello generalmente usato ($p = 0.001$) ha richiesto al mio computer uno sforzo computazionale importante.

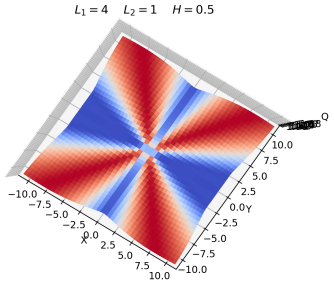


Fig.15: Q-plot

Ho quindi deciso di eliminare un grado di libertà dalla trattazione lasciandola però, per le ragioni sopra descritte, completa. Tale grado di libertà è il raggio.
L'algoritmo è quindi stato eseguito su una circonferenza di raggio fisso $R = 6$ e suddivisa in 100 angoli ϕ diversi. Nel grafico si è quindi rappresentata in ascissa la quantità $\frac{\phi}{\pi}$ e in ordinata il solito valore Q corrispondente al punto (X, Y) determinato dalle coordinate circolari.

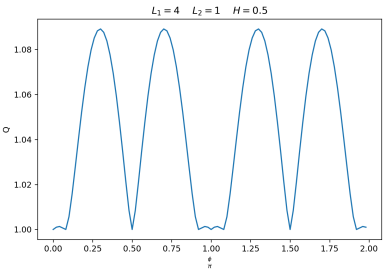


Fig.16: QR-plot

Si può osservare la corrispondenza dei due metodi anche guardando figura 15 e figura 16, dato che tali grafici rappresentano l'algoritmo eseguito nei due diversi modi, ma sulla stessa funzione.

Da questo punto della tesi in poi si è sempre calcolato l'algoritmo secondo quest'ultimo metodo descritto con passo $p = 0.001$.
Ci sarà inoltre utile il valore $< Q >_c$ definito come media tra tutti i valori di Q calcolati nei punti della circonferenza e corrispondenti ad un certo paraboloide

Confronto $H = 0.5$ con $H = 1$

Per trarre delle conclusioni ho considerato una serie di paraboloidi con $L_1 = 10$ e $L_2 = 1$ ruotati di una serie di angoli θ diversi. Nel totale 50 valori equidistanti di θ nell'intervallo che va da 0 a 0.5π (0.5π escluso). In figura 17 è presentato il risultato del nostro calcolo.
Si definisca ora la quantità $< Q >_\theta$ come la media tra tutti i valori di Q calcolati nei punti della circonferenza per tutte le rotazioni θ della funzione considerata.

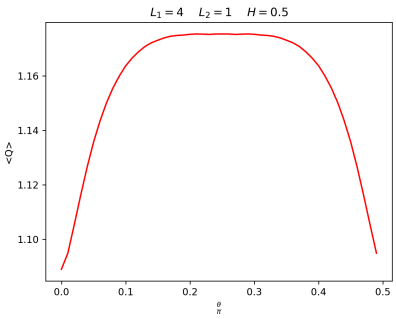


Fig.17: Percorso medio per angolo

Si nota innanzitutto che per $H = 1 < Q >_\theta = < Q >_c$, equazione derivante dal seguente osservazione: la rotazione dei paraboloidi corrisponde, in questo caso, solo ad una rotazione dei percorsi e quindi rende $< Q >_c$ invaiante rispetto a θ.

Ho quindi calcolato per $H = 0.5 < Q >_\theta = 1.16$ e per $H = 1 < Q >_c = 1.15$.

Mediamente, dunque, il nostro campo vettoriale funziona peggio del gradiente nel convogliare verso il minimo di una funzione parabolica.

Metodo rotazionale

Ho cercato di confrontare il gradiente della funzione non ruotata con quello della funzione ruotata di $\theta = 0.25\pi$ in modo da trovare una caratteristica per la distinzione tra un gradiente che trasformato migliora l'algoritmo e uno che trasformato lo peggiora. Ciò che si riscontra è che nel gradiente che migliora, una volta deformato, tanti dei vettori che fanno parte del campo vettoriale sono circa paralleli ad un asse. In quello in cui peggiora, tanti vettori sono invece più o meno paralleli ad una bisettrice.

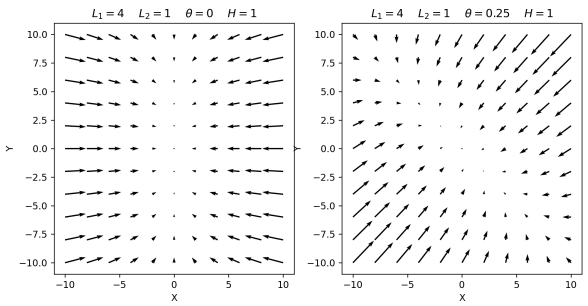


Fig.18: Gradiente delle funzioni

Spiegherò dunque il perché sia conveniente in termini di trasformazione del gradiente avere il primo vettore considerato dall'algoritmo parallelo ad un asse. Mi riferirò nella spiegazione in particolare alla figura di 19 essendo rappresentativa della maggior parte dei percorsi nel caso la funzione sia un paraboloide.
Per ipotesi innanzitutto avremo che la trasformazione del primo vettore non influisca sull'esecuzione dell'algoritmo (ricordando che gli assi sono "punti fissi").

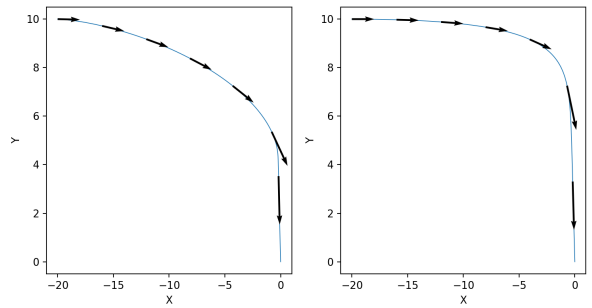


Fig.19: A sinistra il percorso compiuto dall'algoritmo con $H = 0.5$ a destra con $H = 1$

Successivamente ai primi punti però, com'è possibile vedere nella figura del percorso per $H = 1$, i vettori considerati non saranno più esattamente paralleli all'asse X, ma ruotati da tale asse leggermente in senso orario; questo comportamento è prevedibile poiché il minimo si trova in coordinata (0, 0) e man mano che ci si avvicina alla valle il gradiente tenderà sempre più a puntare in tale direzione. Applicare quindi la trasformazione da noi considerata a vettori di questo tipo avrà un'unica conseguenza: li ruoterà in senso orario; si può infatti vedere nella figura del percorso per $H = 0.5$ che i vettori del campo vettoriale formano un angolo più grande con l'asse X di quelli nella figura del percorso per $H = 1$.

Infine prima di arrivare al minimo si ha una fase in cui il gradiente ha un angolo con l'asse X maggiore di 0.25π, come si può vedere dal penultimo vettore nella figura corrispondente. Questa volta la trasformazione del gradiente opererà ruotando il vettore in senso antiorario portandolo, come sempre, verso la bisettrice; il risultato si può vedere esemplificato nella figura del percorso per $H = 0.5$ sempre dal penultimo vettore.

Complessivamente il processo porta una deformazione del percorso compiuto che si può un po' semplicisticamente descrivere come la tendenza a "tagliare le curve", tendenza che chiaramente porta ad un vantaggio computazionale.

Ho a questo punto sviluppato un nuovo algoritmo che sfruttasse tale principio, ne esplichiamo ora il funzionamento.

Supponendo esso parta da una certa coordinata (X, Y), allora calcolerà il gradiente della funzione in tale punto. A seguito di questo il nuovo algoritmo ruoterà gli assi fino ad avere il vettore prima calcolato parallelo all'asse Y (con verso nella direzione positiva delle ordinate). Dunque verrà calcolato $Grad_T$ secondo la formula già specificata ad inizio tesi.
Infine verrà eseguito il vecchio algoritmo in cui $v = Grad_T$ appena calcolato e (X, Y) è il punto di partenza (i passaggi esatti sono descritti nella prefazione della tesi).

Vediamo ora alcuni esempi in cui sono confrontate le diverse modalità di esecuzione dell'algoritmo.

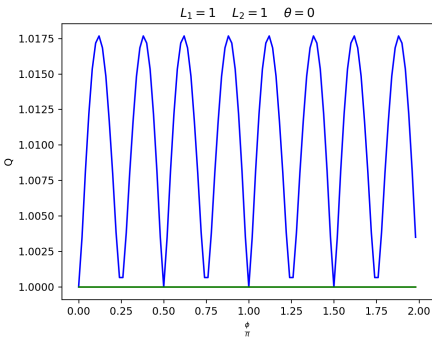


Fig.20: QR-plot

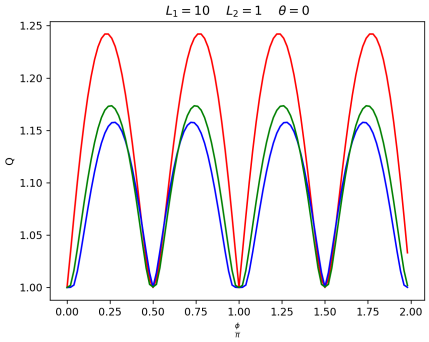


Fig.21: QR-plot

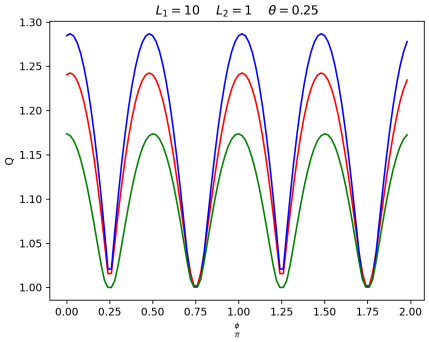


Fig.22: QR-plot

Legenda: i dati in rosso corrispondo all'algoritmo eseguito con $H = 1$, i dati in blu all'algoritmo eseguito con $H = 0.5$, i dati in verde invece corrispondono all'algoritmo eseguito con il metodo appena ideato (della rotazione) con $H = 0.5$. In figura 17 rosso e verde sono sovrapposti.

In figura 20 si osserva che il metodo rotazionale per $L_1 = 1$ $L_2 = 1$ è ideale come quando si ha $H = 1$. Risolve quindi innanzitutto il problema per cui, in questa configurazione, il campo vettoriale trasformato funzionava peggio del gradiente.
Passando invece alla figura 21 ci si accorge che nella configurazione vantaggiosa per $H = 0.5$ si ha che il metodo rotazionale è leggermente peggiore dell'altro, mentre $H = 1$ rimane di gran lunga il peggiore tra tutti.
Infine valutando figura 22 si trova che il metodo rotazionale è il migliore proprio quando il campo vettoriale trasformato con $H = 0.5$ fallisce.

Ora, per gli stessi ragionamenti fatti per $H = 1$, avremo che per il metodo rotazionale vale
$< Q >_\theta = < Q >_c$ indipendentemente dal valore di H.

Ho quindi calcolato per il paraboloide con $L_1 = 10$ $L_2 = 1$ con il metodo rotazionale, impostato $H = 0.5$, il valore $< Q >_c = 1.10$.
Ricordando dunque la computazione con il metodo tradizionale di $< Q >_c = 1.15$ per il campo $H = 1$ e di $< Q >_\theta = 1.16$ per il campo $H = 0.5$, si trae la conclusione che l'algoritmo rotazionale è per ora il modo il migliore di procedere.

Il valore migliore di H

Mi sono quindi chiesto, trovato ora un buon algoritmo, con quale valore di H funzionasse meglio. In questa analisi si è eseguito l'algoritmo per 50 valori di H nell'intervallo tra 0 e 1 ($H = 1$ escluso). Si sono poi normalizzati i valori trovati e mediati sulla circonferenza, così da ottenere un valore medio per ogni valore di H.

Mostriamo a fianco i grafici in cui si ha in ascissa i valori di H e in ordinata il valore medio dei percorsi corrispondente.

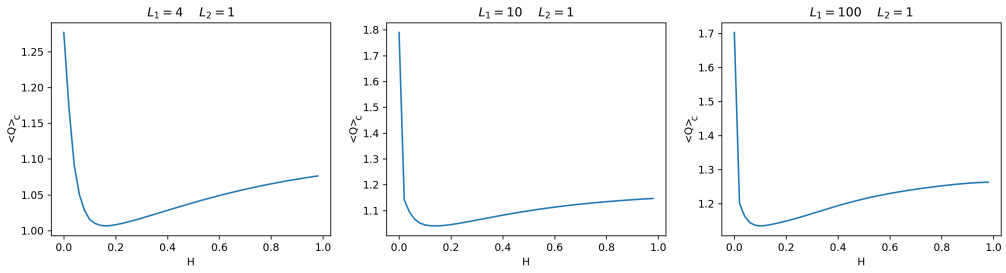


Fig.23

Nonostante il valore ottimale (chiamato H_b), corrispondente al minimo dei grafici in figura 23, sia leggermente diverso al variare della lunghezza della valle, esso si trova circa sempre nella stessa posizione. Possiamo infatti dire che $0 < H_b < 0.2$ e che $H_b \sim 0.14$.

Per capire per quale ragione $H = 0.14$ sia migliore di $H = 0.5$ sono stati mostrati i loro percorsi in figura 24.
In rosso è rappresentato $H = 0.14$ e in blu $H = 0.5$.

Evidenti risultano, innanzitutto, i cambiamenti agli angoli seguenti: $0.25\pi, 0.75\pi, 1.25\pi, 1.75\pi$.

Questi corrispondo ad un abbassamento notevole dei picchi in cui il percorso per raggiungere il minimo è decisamente non ideale. Questo è facilmente spiegabile se consideriamo che minore è H, maggiore è la rotazione del campo vettoriale originario; di conseguenza risulterà che il nostro percorso avrà una tendenza maggiore a muoversi direttamente verso il minimo.
L'altra peculiarità riguarda invece i picchi che si sono venuti a formare agli angoli $0.5\pi, \pi, 1.5\pi$.

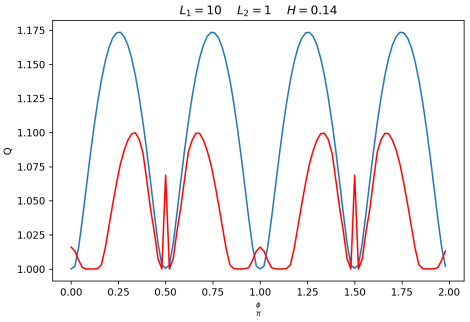


Fig.24: Percorsi di $L_1 = 10$ $L_2 = 1$

Cercando di capire cosa stesse succedendo in tali angoli ho considerato, in figura 25, il percorso compiuto dall'algoritmo partendo proprio dalla coordinata $(0, 6)$. Come è possibile vedere in figura 26 il peggioramento nel calcolo dei percorsi è dovuto alla forma del gradiente trasformato, tale da far proseguire l'algoritmo alternatamente a sinistra e a destra.
Questo deriva dal fatto che il passo $p = 0.001$ abbia un valore finito; guardando infatti il gradiente trasformato della funzione ci accorgiamo che nello spazio intermedio di un salto si trova un vettore parallelo all'asse Y. Questo vuol dire che se il passo fosse davvero infinitesimo ad un certo punto ci troveremmo proprio su tale vettore e da tale vettore continueremmo fino al minimo, proseguendo in maniera rettilinea.

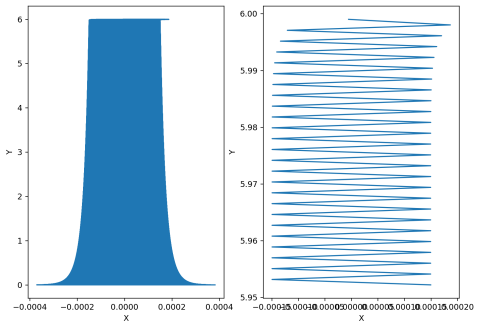


Fig.25: A destra il percorso completo, a sisnistra solo i primi 50 punti considerati

Se fosse possibile la risoluzione di questo problema si avrebbe un ulteriore miglioramento del gradiente trasformato. Questo però richiede lo sviluppo di un algoritmo più complesso, questione che non verrà approfondita in questa tesi.

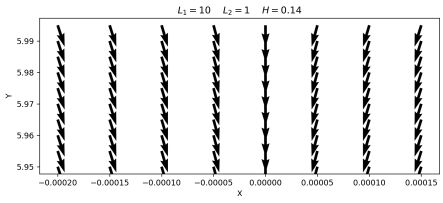


Fig.26: gradiente trasformato

Conclusioni

Come conclusione descriverò quali passi potrebbero essere intrapresi in futuro per una caratterizzazione più ampia del metodo.

In primo luogo, come già accennato, sarebbe interessante comprendere come evitare l'andamento alternato che provoca l'allungamento del percorso, dato che l'origine di questo comportamento è puramente numerica e dovuta al passo finito dell'algoritmo.

Separatamente a ciò, o successivamente, troverei necessario testare l'algoritmo su funzioni diverse dal paraboloide. Nonostante infatti, come già detto, all'intorno del minimo tutte le funzioni si comportino come un paraboloide, distanti da esso potrebbero esserci problemi indesiderati.
In particolare fino ad ora il nostro percorso doveva fare una sola "curva" nel piano XY; perciò potrebbe essere molto interessante cercare di capire cosa accadrebbe se ve ne fossero più di una.
O ancora mentre per quanto riguarda il paraboloide l'algoritmo è invariante di scala, non è detto che ciò si verifichi per funzioni con meno simmetrie di questa.
Infine i risultati ottenuti, per quanto riguarda il valore di H ideale, sono molto stimolanti: pare che $H \sim 0.14$ sia il migliore per tutti i paraboloidi o quasi; varrà la stessa tendenza anche per altre funzioni?

Un ultimo ambito rilevante da approfondire potrebbe essere il caso in cui sia $H > 1$. Come detto ad inizio tesi una trasformazione di questo tipo avrebbe tendenze opposte rispetto alla rotazione studiata fino ad adesso, tuttavia questo non preclude l'utilità della stessa. Infatti, nel caso in cui la funzione sia ruotata di $\theta = 0.25\pi$, una trasformazione con $H > 1$ potrebbe dare gli stessi vantaggi di quella con $0 \leq H < 1$ per quanto riguarda la funzione non ruotata.
Si potrebbe quindi produrre una trattazione come quella appena presentata, considerando la trasformazione citata in questo paragrafo.

Tali approfondimenti sono quindi quanto mi aspetterei dai prossimi studi sull'argomento. Concludo augurando un buon lavoro a chiunque voglia intraprendere questa ricerca.

Bibliografia

[1] M. Baiesi, Power Gradient Descent (2019)